кількості змінних, а відповідно, і зменшенням об'єму пам'яті, яка використовується для роботи програми. На скорочення часу обробки запитів вплинуло і винесення функцій по обробці винятків в окремі модулі. Зменшився загальний відсоток дублювання коду, а відповідно, і затрати системи на його виконання.

Висновок

Варто відмітити, що суттєве скорочення коду відбулось на сервісних рівнях та рівні контролера. Судячи з цього можна сказати, що впровадження АОП не впливає на структури даних, якими оперує програма, воно тільки скорочує процедуру їх обробки та бізнес-логіку програми. Той факт, що впровадження АОП не вплинуло на структури даних,свідчить і про те, що використання АОП тільки доповнює ООП - програму, та не унеможливлює використання принципів ООП.

Список використаних джерел:

1. Spring Faremework Documentation, [Електронний ресурс]. – Режим доступу: http://docs.spring.io/.
2. AOP@Work: Мифы и реальности АОП, [Електронний ресурс]. – Режим доступу: https://www.ibm.com/.
3. Аспектно-Ориентированное Программирование в Spring, [Електронний ресурс].
4. Знакомство с АОП, [Електронний ресурс]. – Режим доступу: http://habrahabr.ru/post/114649/.
5. Г. Буч. Объектно-ориентированный анализ и проектирование / Г. Буч. – СПб.: Издательство Бином, Невский диалект, 1998. – 560 с.

УДК 004.4

ІНЖЕНЕРІЯ МЕТОДІВ РОЗРОБКИ ВЕБ-ДОДАТКІВ ЗАСОБАМИ JAVA STANDART EDITION

Гончар Л.І.[1)], Кохан І.І. [2)]

Тернопільський національний економічний університет

[1)] к.е.н., доцент; [2)] магістрант

Недоліком використання Java Enterprise Edition є залежність від великих фреймворків, в яких закладено набагато більше логіки, ніж потрібно для конкретного програмного продукту. Взаємодія цих фреймворків із сторонніми бібліотеками та з модулями додатку значно ускладнює програмній продукт.

Для того щоб в програмному коді було легше знайти бізнес-логіку, розробка повинна відштовхуватись від області застосування додатку.

Використання веб-фреймворків не є обов'язковим для Java. Натомість слід розглядати можливість створення простих додатків методами Java Standard Edition [2].

Проблема розгортки проекту на сервері може бути вирішена вбудованим сервером, який буде запускати проект із Java-коду. Прикладом такого сервера є Jetty. Jetty може бути запущений локально (для розробки або у дрібномасштабному виробництві) або як частина іншого HTTP сервера для повномасштабної роботи. Використання Maven, Gradle, та інших інструментів для збірки проекту не завжди виправдане. Невеликі проекти доцільніше збирати напряму за допомогою Ant, а їх залежності дуже легко регулюються Ivy. Це може зменшити кількість конфігураційних файлів та умов для розгортки додатку до мінімума.

Таким чином, не використовуючи функціонал JEE, лише за допомогою Java Standard Edition, можна створити повноцінний веб-проект, розробка і подальша підтримка якого буде набагато ефективніша по часу і вартості продукту.

У науковій роботі було проведено порівняльне дослідження статистичних характеристик програми з ідентичним до предмету дослідження фунціоналом. Важливо відмітити, що набір функцій програми не змінювався, дві версії виконують один і той самий набір функцій. Відмінністю другої версії є відмова від використання Spring MVC, JSP, ESB та SOAP веб-сервісів.

Таблиця 1

Порівняння кількісних характеристик коду програми з застосування класичного підходу та програми без використання веб-фреймворків.

	Кількість рядків коду	Кількість класів	Кількість Логічних рядків*	Логічних рядків (%)	Дублювання коду (%)**
Spring MVC	1891	90	75	3.96	18
Без каркасів	204	12	63	25	0

*Логічні рядки – рядки коду, які виконують завдання бізнес-логіки.

**Аналізу дубльованого коду підлягають лише java-файли. Дескриптори xml, сторінки html, css та js файли не підлягають такому аналізу.

Виходячи із порівняльної оцінки (табл. 1) коду двох програм з однаковим функціоналом але різними підходами, можна сказати, що відмова від використання громіздких конструкцій веб-фреймворків суттєво покращує характеристики коду. При використанні Java Standard Edition і простої архітектури, спостерігається зменшення об'єму коду в 9.2 рази, зменшення кількості класів на 86.7%, повне зникнення дубльованого коду, збільшення відношення логічних рядків до всього коду до 25% (в 6.3 рази) [5].

За сукупністю кількісних характеристик видно, що в цілому вдалося позбутися 89% коду, не втративши функціонал. Такий високий відсоток доводить, що використання веб-фреймворків не є доцільним у проектах малого розміру. Запропонований підхід вирішує основний недолік веб-додатків, написаних мовою Java, а саме складність їх архітектури і малу частину логічного коду.

Сучасний веб-додаток мовою Java, незалежно від його складності, є багатошаровою конструкцією, в якій відведено вкрай мало місця для бізнес-логіки. Більшу частину архітектури проекту займає реалізація фреймворків, створення зв'язків між шарами додатку, організація взаємодії модулів проекту. На відміну від поширеної думки, використання веб-фреймворків не є обов'язковим, а для малих проектів виявляється надлишковим та негативно впливає на кінцевий результат.

Використання комплексу сучасних рішень, таких як RESTful веб-сервіси у поєднані з інтерфейсом на JavaScript дає можливість спростити веб-додатки та значно прискорити їх розробку.

У результаті проведених досліджень з використанням такого підходу встановлено, що метрики коду для малих проектів майже в десять разів нижче, ніж в класичній реалізації. Зафіксовано покращення таких кількісних характеристик як: зменшення об'єму коду та складності коду, кількості змінних та класів. Також спостерігається покращення архітектури програми, так як бізнес-логіка більше не завуальована нефункціональним кодом для підтримки архітектури,

Висновок

Існуючі інструменти розробки на мові Java поза Java Enterprise Edition надають всі можливості та механізми для створення простих і прозорих програмних продуктів та є добре документованими і зручними у використанні.

Враховуючи сукупність факторів, є цілком можливим, що наступні редакції Java не будуть нав'язувати конструкції веб-фреймворків за замовчуванням, а розширять необхідний розробнику функціонал для створення простих Java-додатків для веб.

Список використаних джерел

1. Как нам спасти Java?, [Електронний ресурс]. – Режим доступу: http://gotocon.com/
2. Шилд Г. Полный справочник по Java Standard Edition 6 / Шилд Г. – Издательский дом «Вильямс», 2007 – 1040 с.
3. Spring Faremework Documentation, [Електронний ресурс]. – Режим доступу: http://docs.spring.io/.
4. AOP@Work: Мифы и реальности АОП, [Електронний ресурс]. – Режим доступу: https://www.ibm.com/developerworks/ru/library/j-aopwork15/.
5. Г. Буч. Объектно-ориентированный анализ и проектирование / Г. Буч – СПб.: Издательство Бином, Невский диалект, 1998. – 560 с.
6. Статистичний аналіз коду програм на Java, [Електронний ресурс]. –Режим доступу: http://www.sonarqube.org/.